APPLICATION FOR UNITED STATES LETTERS PATENT

FOR

RULE BASED PACKET PROCESSING ENGINE

Inventor(s): Ravi L. Sahita
Priya Rajagopal

Prepared by: Justin B. Scout,
Reg. No. 54, 431

intel®
Intel Corporation

"Express Mail" label number: EV325528763US

RULE BASED PACKET PROCESSING ENGINE

BACKGROUND

1. Field

The present disclosure relates to the processing of a packet utilizing a generic rule based engine, and, more specifically, to the processing of a packet or stream of packets, which are transmitted across a network, utilizing a generic rule based engine that is part of a network processor.

2. Background Information

A packet is generally a unit of information typically transmitted as a whole from one part of a network, a source, to another part of a network, a destination. These packets may or may not be of a fixed length. A series, flow or stream of packets taken together may constitute a complete transmission of information across the network.

As packets flow through a network, they may be inspected and processed by various devices along the path from the source to the destination. Packet inspection at an intermediate node is a common part of the network environment. In this environment the need to treat, or process, packets differently is often considered critical in order to ensure the desired quality of service (QoS) and performance requirements of the network to satisfy users. Also, in many cases, security systems, such as, for example, firewalls and intrusion detection services (IDS), frequently inspect packets to detect virus patterns and enforce security policies. Ideally this type of packet processing should have minimal impact on the performance of the network.

A packet is frequently processed using a rule. A "rule" is, in this context, a combination of a set of conditions and associated actions to occur if the conditions are satisfied or met. As rules increase in complexity, more processing is required and the impact on the performance of the network increases. A need therefore exists to improve the efficiency of the processing of rule based packet processing.

BRIEF DESCRIPTION OF THE DRAWINGS

Subject matter is particularly pointed out and distinctly claimed in the concluding portions of the specification. The disclosed subject matter, however, both as to organization and the method of operation, together with objects, features and advantages thereof, may be best understood by a reference to the following detailed description when read with the accompanying drawings in which:

FIG. 1 is a flowchart illustrating an embodiment of a technique for the processing of a packet utilizing a generic rule based engine in accordance with the disclosed subject matter;

FIG. 2 is a block diagram illustrating an embodiment of an apparatus or memory structures utilized by a technique for the processing of a packet utilizing a generic rule based engine in accordance with the disclosed subject matter;

FIG. 3 is a block diagram illustrating an embodiment of an apparatus or memory structures utilized by a technique for the processing of a packet utilizing a generic rule based engine in accordance with the disclosed subject matter; and

FIG. 4 is a block diagram illustrating an embodiment of a system and apparatus that allows for the processing of a packet utilizing a generic rule based engine in accordance with the disclosed subject matter.

DETAILED DESCRIPTION

In the following detailed description, numerous details are set forth in order to provide a thorough understanding of the present disclosed subject matter. However, it will be understood by those skilled in the art that the disclosed subject matter may be practiced without these specific details. In other instances, well-known methods, procedures, components, and circuits have not been described in detail so as to not obscure the disclosed subject matter.

FIG. 1 is a flowchart illustrating an embodiment of a technique for the processing of a packet utilizing a generic rule based engine in accordance with the disclosed subject matter. Block 110 illustrates that a network processor or other device may receive a packet, or flow of packets. In this context, a network processor is a device that inspects and processes packets as they flow across a network. It is contemplated that a network processor may be a specific device, such as, for example, an Intel® Exchange Architecture network processor. Alternatively, in another embodiment, a network processor may be a more generic-use processor configured to act, in at least part, as a network processor. Also, it is contemplated that in one embodiment, the network processor may be combination of devices.

Block 120 illustrates that the packet may be checked to determine if the Active Rule applies to the packet. In one embodiment, the most recently used rule may be cached and considered the Active Rule. In other embodiments, other criteria for establishing the Active Rule may be used. Block 130 illustrates that if the Active Rule is not applicable to the packet, a cached Rules Table may be consulted to determine if any rule applies to the received packet. Block 135 illustrates that, if a rule is applicable, then the Applicable Rule may be made the Active Rule. It is contemplated, that, in one embodiment, Block 120 may be skipped and Block 130 immediately executed. It is also contemplated that, in one embodiment, a packet may have at most one rule which is applicable. In another embodiment, multiple rules may be applicable to the received packet. In such an embodiment, a technique of Fig. 1 may be repeated for each rule.

In one embodiment, the applicability of a rule may be determined by utilizing a Rules Table that allows relatively quick comparison to a received packet, as illustrated by Fig. 2. FIG. 2 is a block diagram illustrating an embodiment of an apparatus or memory structures utilized by a technique for the processing of a packet utilizing a generic rule based engine in accordance with the disclosed subject matter. Rules Table 200 may includes a number of rules, or as illustrated abbreviated rules (hereafter "rules"), such as rules 210, 220, & 230. In one embodiment, a rule, for example rule 210, may include all the conditions and actions needed to process the rule. In another embodiment, the Rule Table may only contain rules with enough information to allow the applicability of the rule to be determined by a pointer to the full conditions and actions of the rule. For example, rule 210 only includes a source field 212, destination field 214, protocol field

216, and a rule pointer 218. Rules 220 & 230 includes similar respective fields. However, it is contemplated that other embodiments may include other fields.

In the embodiment illustrated by Fig. 2, a received packet may be considered applicable to rule 210 if the received packet originated from the source of field 212, is being transmitted to the destination of field 214, and utilizes protocol 216. Likewise, rules 220 & 230 may be considered applicable if their respective fields match the received packet. It is contemplated that in other embodiments of the disclosed subject matter, other criteria for rule applicability may be used. In one embodiment, the Rule Table 200 may be stored within a quickly accessible local memory, for example a Content Addressable Memory (CAM).

In one embodiment, once a rule is determined to be applicable, the rest of the rule may be accessed by looking up the rule's conditions and actions in a Rule Group Table 201. For example, if rule 210 is determined to be applicable to the received packet, the Rule Pointer field 218 may point to Rule Group 240 that contains, or at least facilitates access to the rule's conditions and actions. In the embodiment illustrated by Fig. 2, Rule Group 240 may include a field 241 containing the number of condition sets associated with rule 210, a field 243 containing a pointer to the first condition set, a field 245 containing the number of actions associated with rule 210 and, a field 247 containing a pointer to the first action set. Rule Groups 250, 260, 270, 280, & 290 would include similar respective fields. However, it is contemplated that other embodiments, may integrate the information of the Rule Group table directly within the Rule Table 200 or divide the Rule Group Table into Condition and Action Tables, or other organization. It is contemplated that, in one embodiment, the Rule Group Table may be maintained in a memory structure that is less rapid than the Rule Table.

Block 140 of Fig. 1 illustrates that once an Active Rule has been identified, a Condition Set Table associated with the Active Rule may be accessed. The Condition Set Table may allow access to the Condition Sets of the rule. Fig. 3 illustrates one embodiment of a Condition Set Table.

FIG. 3 is a block diagram illustrating an embodiment of an apparatus or memory structures utilized by a technique for the processing of a packet utilizing a generic rule based engine in accordance with the disclosed subject matter. Rule Group 240 may be associated with the Active Rule and include a field 243 that points to the first Condition Set or, in another embodiment, Condition Set Table associated with the Active Rule. In this embodiment, Condition Set Table 310, includes two Condition Sets 313 & 316. Each Condition Set 313 & 316 may include a field denoting the number of Conditions in each Set and a pointer to the first Condition in the respective sets.

In the illustrated embodiment of Fig. 3, the pointer of the Condition Set 310 points to a Condition Indirection Table 320 that holds pointers to the conditions. The Condition Indirection Table may be used to facilitate access to the Conditions as multiple Condition Sets may share identical Conditions. In one embodiment, the Condition Indirection Table may allow the Conditions of a Condition Set to be processed sequentially, but accessed randomly. The Condition Indirection Table may include Condition Set Pointer 330 that contains the pointers for Condition Set 313, and Condition Set Pointer 340 that contains the pointers for Condition Set 316. Of course, other embodiments of the disclosed subject matter are contemplated and this is merely an illustrative embodiment.

In one embodiment, the Condition Set Table or a portion of it may be cached within a Static Random Access Memory (SRAM). In a specific embodiment, the number of total Conditions Sets may be limited by the number of Conditions Sets that may be read in one clock cycle, for example 8, such as is the case for the Intel® IXP2400. However, other embodiments that do not have a limitation on the number of Condition Sets used are contemplated and within the scope of the disclosed subject matter.

Block 150 of Fig. 1 illustrates that for each Condition Set, blocks 160, 170, 180, and 190 may occur. Block 160 illustrates that each Condition of the Active Condition Set may be processed. Block 165 illustrates that if each Condition in the Condition Set is met the Condition Set will be considered to be met. In one embodiment, each Condition is evaluated to obtain a Boolean ("true" or "false") value. These values may then be logically ANDed together to establish a Boolean value for the Condition Set. In one embodiment, a 1-bit Boolean Condition Accumulator may be used to store the cumulative result of the Condition Set evaluation, which each Condition result being ANDed against the Condition Accumulator as the Condition is evaluated. It is contemplated the other embodiments may logically OR the Conditions or utilize a more complex evaluation scheme.

Fig. 3 illustrates an embodiment of the Condition Sets and Conditions. In one embodiment, Condition Table 350 may store the Conditions 351, 353, 355, & 357. Condition Set 313 may include two Conditions 351 & 353. In the illustrated embodiment, the Conditions are accessed by accessing Condition Set Pointers 330 that uses Condition Pointer 333 to point to Condition 351, and Condition Pointer 336 to point to Condition 353. Condition Set 316 includes three Conditions 351, 355, & 357. In the illustrated embodiment the conditions are accessed utilizing Condition Set Pointers 340 and Condition Pointers 343, 346, & 349, similarly to as just described. However, other implementations and embodiments are contemplated and are within the scope of the disclosed subject matter.

In one embodiment, a Condition, such as, for example Condition 351, may be based on examining the received packet for the presence of a particular pattern. For example, a particular virus pattern could be examined; however, this is merely an illustrative example. To facilitate this examination, the Condition may include, in one embodiment, fields such as, for example, the bit offset at which the pattern is expected to occur, the pattern itself, an opcode denoting the form of examination (*e.g.* equal to, not equal to, greater than, *etc.*), a pattern mask to modify the pattern, and an mask operation.

In one embodiment, the Condition may include a Save State Flag that may denote whether or not the Condition has been evaluated for the received packet, and the state of that evaluation. In the illustrated embodiment of Fig. 3, Condition 351 is part of two Condition Sets and therefore would be executed twice. A flag may be used to prevent the needless second execution and return the cached result of the first execution.

In one embodiment, the Condition Table or a portion of it may be cached within a local memory, possibly a Content Addressable Memory (CAM). In a specific embodiment, the number of total Conditions may be limited by the number of entries in the CAM, for example 16, such as is the case for the Intel® IXP2400. However, other embodiments that do not have a limitation on the number of Conditions used are contemplated and within the scope of the disclosed subject matter.

Block 170 of Fig. 1 illustrates that if any of the Conditions Sets were met the Actions Set of the Rule will be performed. As described above, in reference to Block 160, if each Condition in the Condition Set is met the Condition Set will be considered to be met. In one embodiment, each Condition Set is evaluated to obtain a Boolean value. These values may then be logically ORed together to establish a Boolean value for the Rule. This is in contrast to the Conditions that were ANDed in order to establish a Boolean value for the Condition Sets. In one embodiment, a 1-bit Boolean Condition Set Accumulator may be used to store the cumulative result of the Rule evaluation, which each Condition Set result being ORed against the Condition Set Accumulator as the Condition Set is evaluated. It is contemplated the other embodiments may logically AND the Condition Sets or utilize a more complex evaluation scheme.

Block 180 illustrates that, if the Active Rule was met, an Action Set associated with the Active Rule may be accessed. The Action Set may include a number of Actions that are to be preformed on or because of the received packet. It is contemplated that, in one embodiment, the Action Sets may be stored within an Action Set Table. In one embodiment, the Action Set Table may be stored within an SRAM. In a specific embodiment, the number of total Actions within an Action Set may be limited by the number of entries that can be read in one clock cycle, for example 16 32-bit values, such as is the case for the Intel® IXP2400. However, other embodiments that do not have a limitation on the number of Actions used are contemplated and within the scope of the disclosed subject matter.

Block 190 illustrates that each Action in the Active Rule's Action Set may be executed. An action may include things, such as, for example, modifying the packet, providing the packet with high priority throughput, deleting the packet, generating a second packet (possibly to report to an Intrusion Detection System), or report an error; however, these are merely a few illustrative examples to which the disclosed subject matter is not limited.

In one embodiment, the Actions may return a value denoting successful completion or other status. In another embodiment, the Actions in the Actions Set may be chained together and executed sequentially. In another embodiment, the Actions may be executed in substantially simultaneously, or a combination of sequentially and simultaneously.

Block 195 illustrates that the receive packet may be forwarded to its destination or next intermediate node or next packet processing system component. In one embodiment, this block may not be performed if the packet was sufficiently modified or deleted by the Actions of Block 190. In one embodiment, Block 195 may be performed if the Rule was not met in Block 170.

FIG. 4 is a block diagram illustrating an embodiment of a system and apparatus 401 that allows for the processing of a packet utilizing a generic rule based engine in accordance with the disclosed subject matter. In one embodiment, apparatus 401 may include a Network Processor Core 410, a Packet Buffer 460, a cache memory 420, and a Micro-Engine 470. In another embodiment, the apparatus may include a plurality of Micro-Engines operating substantially simultaneously on a plurality of received packets.

Network Processor core 410 may be capable of resource management and receiving a program and control logic from a source external to the apparatus. Packet buffer 460 may be capable of storing and buffering a packet. In one embodiment, once a packet is fully received the packet buffer may release the packet to Packet Processing Engine (PPE) Ingress 473. In another embodiment, the Packet Buffer may copy the packet to the PPE Ingress. In one embodiment, the Packet Buffer may accessible to the plurality of Micro-Engines. In one embodiment, the Packet Buffer mày include Dynamic Random Access Memory (DRAM).

In one embodiment, Micro-Engine 470 may include a Packet Processing Engine (PPE) Ingress 473 that is capable of receiving a packet. In one embodiment, the PPE Ingress may also be capable to evaluating the received packet to determine if the Active Rule is applicable; however, in another embodiment, the PPE Ingress may merely be a storage location. The Micro-engine may also include a Rule Based Action PPE 475. The Rule Based Action PPE may be capable of performing a technique as illustrated by Fig. 1 and discussed above. In one embodiment, the Rule Based Action PPE may include a local memory and a Content Addressable Memory (CAM) that is capable of storing the memory structures described above. The Micro-engine may also include a PPE Egress 478 that is capable to storing a packet that has been processed by the Rule Based Action PPE.

In one embodiment, the cache memory 420 may include data structures Condition Set 430, and Action Set 440 which are described in detail above. In one embodiment, the Micro-Engine 470 may have access to the cache memory. In one embodiment, the plurality of Micro-Engines may all have access to the cache memory and the data structures includes within. In one embodiment, the cache memory may also include a Packet Stream Buffer 450 that is capable to storing information and, in some embodiments, previous packets from the same packet stream as the received packet. In one embodiment the cache memory may include SRAM.

The techniques described herein are not limited to any particular hardware or software configuration; they may find applicability in any computing or processing environment. The techniques may be implemented in hardware, software, firmware or a combination thereof. The techniques may be implemented in programs executing on programmable machines such as mobile or stationary computers, personal digital assistants, and similar devices that each include a processor, a storage medium readable or accessible by the processor (including volatile and non-volatile memory and/or storage elements), at least one input device, and one or more output devices. Program code is applied to the data entered using the input device to perform the functions described and to generate output information. The output information may be applied to one or more output devices.

Each program may be implemented in a high level procedural or object oriented programming language to communicate with a processing system. However, programs may be implemented in assembly or machine language, if desired. In any case, the language may be compiled or interpreted.

Each such program may be stored on a storage medium or device, *e.g.* compact disk read only memory (CD-ROM), digital versatile disk (DVD), hard disk, firmware, non-volatile memory, magnetic disk or similar medium or device, that is readable by a general or special purpose programmable machine for configuring and operating the machine when the storage medium or device is read by the computer to perform the procedures described herein. The system may also be considered to be implemented as a machine-readable or accessible storage medium, configured with a program, where the storage medium so configured causes a machine to operate in a specific manner. Other embodiments are within the scope of the following claims.

While certain features of the disclosed subject matter have been illustrated and described herein, many modifications, substitutions, changes, and equivalents will now occur to those skilled in the art. It is, therefore, to be understood that the appended claims are intended to cover all such modifications and changes that fall within the true spirit of the disclosed subject matter.